BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

TEXT TO AUDIO ALIGNMENT

SYNCHRONIZACE TEXTU A AUDIA

BACHELOR'S THESIS
BAKALÁŘSKÁ PRÁCE

AUTHOR **TOMÁŠ ŠÍMA**
AUTOR PRÁCE

SUPERVISOR **Ing. IGOR SZŐKE, Ph.D.**
VEDOUCÍ PRÁCE

BRNO 2017

Zadání bakalářské práce/20997/2017/xsimat00

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2017/2018

Zadání bakalářské práce

Řešitel: **Šíma Tomáš**
Obor: Informační technologie
Téma: **Synchronizace textu a audia**
Text to Audio Alignment
Kategorie: Zpracování řeči a přirozeného jazyka

Pokyny:

1. Nastudujte principy automatického zarovnání textu k audiu.
2. Navrhněte postupy, jak automaticky zarovnat text s odpovídajícím audiem. Uvažujte i postupy nezávislé na jazyku.
3. Implementujte navržené postupy. Zarovnejte vybraná data a porovnejte úspěšnost jednotlivých postupů.
4. Identifikujte místa (příčiny) selhání navržených postupů a pokuste se je vylepšit.
5. Znodnoťte dosažené výsledky a navrhněte směry dalšího vývoje.
6. Vyrobte A2 plakátek a cca 30 vteřinové video prezentující výsledky Vaší práce.

Literatura:

- Dle pokynů vedoucího

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Szőke Igor, Ing., Ph.D.**, UPGM FIT VUT
Datum zadání: 1. listopadu 2017
Datum odevzdání: 16. května 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
L.S612 66 Brno, Božetěchova 2

doc. Dr. Ing. Jan Černocký
vedoucí ústavu

Abstract

The purpose of this work is to research existing text-to-speech aligning algorithms. We chose an implementation of one these algorithms, based on Hidden-Markov Joint-Sequence Models, and we explored its strengths, quirks and weaknesses. We explored whether it is possible to predict the alignment accuracy using probability values generated from Viterbi algorithm and the beam search value. Our testing data comes from the BBC as part of MGB Challenge 2015. This data creates, with its high content diversity, near perfect testing set to prove our algorithm is flexible and error independent.

Abstrakt

Účelem této práce je průzkum exitujících algoritmů pro synchronizaci textu a audia. Vybrali jsme exitující implementaci jednoho z těchto algoritmů, který je založený na skrytých markovových modelech sdružených sekvencí a prozkoumaly jsme jeho výhody, nevýhody a podivnosti. Dále jsme ověřili, zda je možné předvídat úspěšnost zarovnání z hodnot generovaných Viterbi algoritmem a hodnotou paprsku. Naše testovací data pochází od BBC a byla součástí MGB Challenge 2015. Díky svojí různorodosti poskytují tato data ideální testovací set k ověření flexibility našeho algoritmu a jakožto i jeho schopnosti tolerovat chyby.

Keywords

Text to Audio Alignment, natural language, MGB Challenge, PhnRec, G2P, Viterbi, Hidden Markov Models, Joint-Sequence Models

Klíčová slova

Synchronizace textu a audia, přirozený jazyk, MGB Challenge, PhnRec, G2P, Viterbi, Skryté markovovi modely, Modely sdružených sekvencí

Reference

ŠÍMA, Tomáš. *Text to Audio Alignment.* Brno, 2017. Bachelor's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Ing. Igor Szőke, Ph.D.

Text to Audio Alignment

Declaration

I hereby declare that this bachelor's thesis was prepared as an original author's work under the supervision of Mr. Igor Szőke, Ph.D. All the relevant information sources, which were used during preparation of this thesis, are properly cited and included in the list of references.

Tomáš Šíma

May 9, 2018

Acknowledgements

Here I would like to express thanks to the supervisor Mr. Igor Szőke, Ph.D. for his endless patience. Without him it wouldn't be possible for this work to exists.

Contents

Chapter 1

Introduction

This work aims to combine speech and written text into one perfectly aligned form, it aims to determine exactly what sound corresponds to what letter in any given time in a way that is language independent. There is a wide range of applications for precise text and audio aligner, whether it is matching subtitles to a movie, a university lecture, or a conference. As mentioned, our solution is language independent, that makes it unique and, I dare to say, near revolutionary in the field of audio science. This language independence means our application is the key solution to anyone wanting to study new languages, such as myself, and that even includes less well-known languages for which there simply is not enough data to successfully train existing aligners and languages where a very precise phoneme derivation method has not yet been developed. As someone who is already proficient in several languages, this completely remarkable property is what motivated me to select the topic and to work on this thesis. I, together with my supervisor, am proud to present the findings of our collaborative work and we hope our findings will be of use to any future readers, researchers and scholars.

Chapter 2

Definitions

In this chapter some lesser known words are defined, basic ideas and thought processes outlined. We describe what do we mean when we say the very word "alignment" and what does it mean in the context of our work. We try to explain why text-to-speech alignment is, without any doubt, such an important task in the modern day and age. We briefly talk about the source of our data, the Multi-Genre Broadcast Challenge. We describe what it is, and we hint at the process of how we use its scoring algorithms to evaluate the quality of our text-to-speech alignment solution. We describe what values we are looking for and what is their exact meaning in the context of this work.

2.1 Alignment

Before we begin to describe anything, we should explain what the word alignment actually means. Should one look into the dictionary one would discover the word alignment is often described as arrangement in correct relative positions [28]. Whenever we are using the word alignment we are, of course, using it in the context of text to speech alignment. This work understands alignment as the 0 to N union between sets of text (graphemes) and sounds (phonemes).

2.1.1 The Gravity Of Text To Speech Alignment

To better understand the gravity of text to speech alignment, please imagine you are in a cinema watching a movie with subtitles. We dare to suggest, you would not enjoy the movie as much if the subtitles were ahead or behind what was happening right there on the screen. That is why subtitles need to be precisely aligned. This laborious task is often done by people. Every TV show, every movie has to have a person manually matching (aligning) subtitles to audio. One of the methods this tedious process is often done, is by having a person repeat everything that is being said on the screen into a computer at the same time as it said on the screen. This is done because phoneme recognizers do not work as perfectly with background audio, now almost omnipresent in nearly all movies and TV shows. This was also one of the reasons why we set out to make our aligner error-independent. When we consider the amount of man-hours required to produce subtitles for all the movies, all the TV stations that run 24 hours a week, it comes as no surprise that there have been many efforts to automate this task and move the alignment process onto computers.

2.1.2 Text To Speech Alignment Real-world Applications

There is a wide range of modern applications to text-to-speech alignment. Every time you see subtitles somewhere, they had to have been aligned first:

- Movie dialogs
- Commentaries
- Television programs
- Video Games

Without subtitles, the viewers who are deaf or hard of hearing couldn't enjoy many of the above mentioned media.

2.2 Boundary

This word is, in the text of this work often abbreviated as bound (short for boundary). Boundary is a value in milliseconds which defines the delta of what is still considered to be correctly aligned data. As an example, "bound of 1000" means we treat the result as correct if the time is distant no less than one second from the actual time.

2.3 The Basic Outline Of Three Commonly Used Units

In this work, two very uncommon words are being used: grapheme and phoneme. In this section we describe their meaning in case a reader is not yet familiar with these terms.

2.3.1 Grapheme

Grapheme is the smallest unit of writing [28]. The word grapheme in the context of this work can be loosely interpreted as 'letter'.

2.3.2 Phoneme

Phoneme is the smallest unit of speech [28]. Phonemes are unique sounds from which speech is composed. Interestingly enough, silence is also treated a type of phoneme (denoted sil).

2.3.3 Graphone

Any number of graphemes mapped to any number of phonemes is called a graphone [13]. The following article [6] has explored this topic in depth and found a simple "0..1 to 0..1" matching to be the optimal solution (as more complicated solutions didn't offer much advantage). Below 2.1 is an image of "0..1 to 0..1" graphone.

m	i	x	-	i	n	g
[m]	[ɪ]	[k]	[s]	[ɪ]	-	[ŋ]

Table 2.1: Graphone alignment (0..1-to-0..1) for word 'mixing', taken from [13]

2.4 Multi-Genre Broadcast Challenge

Our testing data comes from the Multi-Genre Broadcast Challenge competition. For the purposes of evaluating our results, we use Multi-Genre Broadcast Challenge evaluation scripts. In these scripts we focus on three values: Precision, Recall and F-score, to measure the quality of our alignment. The purpose of the Multi-Genre Broadcast Challenge (MGB) is, as they state on their website, the "evaluation of speech recognition, speaker diarization, dialect detection and lightly supervised alignment using TV recordings in English and Arabic" [4]. In this work we are using data from MGB 2015. In the year 2015 MGB had the following challenges [5]:

1. Speech-to-text transcription of broadcast audio
2. Alignment of broadcast audio to subtitles
3. Longitudinal speech-to-text transcription of a sequence of episodes from the same series
4. Longitudinal speaker diarization and linking, requiring the identification of speakers across multiple recordings

Our work is most related to the second task of "alignment of broadcast audio to subtitles". The Multi-Genre Broadcast Challenge has successfully run again in 2016, and while a third run is planned, it hasn't happened yet. This time a different set of input data will be used due to licensing issues with the British Broadcasting Corporation which no longer wishes to share their data [4].

2.4.1 Precision

The terms precision, along with recall and f-score, have been first defined in the context of information retrieval in this article from 1955 [23]. Precision is the answer to the question of how many selected items are relevant. For example, if we were to select all answers we would cover all relevant answers, but this would get us no closer to knowing which answers the relevant ones are. Precision is defined using this formula [23]:

$$precision = \frac{|\{relevant \quad documents\} \cap \{retrieved \quad documents\}|}{|\{retrieved \quad documents\}|}$$

For the purposed of this work, treat "documents" as "words correctly aligned within boundary".

2.4.2 Recall

Recall measures how many relevant items are selected (regardless if we selected just them or them and everything else, which means if we selected all answers we would get a recall value of 1). Recall has been defined in the context of information retrieval [23] as:

$$recall = \frac{|\{relevant \quad documents\} \cap \{retrieved \quad documents\}|}{|\{retrieved \quad documents\}|}$$

As previously mentioned the word "documents" stands for "words correctly aligned within boundary".

2.4.3 F-Score

F-score is more accurately called F_1 score. It connects Precision and Recall into one value between 0 and 1. According to this [7] source: F_n-score "measures the effectiveness of retrieval with respect to a user who attaches n times as much importance to recall as precision". Our script calculates F-Score (F_1) using the following formula:

$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

Whereas a generalized F_n is calculated thus:

$$F_n = (1 + n^2) * \frac{precision * recall}{(n^2 * precision) + recall}$$

2.5 Beam

When the word "beam" is used in this work, it can have two meanings.

2.5.1 Beam Width

The first meaning of the word beam relates to dynamic pruning (often used by Hidden Markov Models) and could be explained as the maximum distance when exploring a graph by expanding the most promising node in a limited set. When the word beam is used in this first meaning, we could basically call it the cut off point for beam search (or beam width for short).

2.5.2 Beam Distance

When we use beam in its second meaning it is almost always in relation to the output of our G2P aligner. This value can reach anywhere between negative beam width and positive beam width and it signifies the current distance when exploring graph by expanding the promising node. The greater the number the greater the distance. The plus or minus sign represent the direction traveled (there is no such a thing as negative distance).

Chapter 3

Possible Approaches To Text-to-speech Alignment

One of the task set for me when writing this work was to explore existing solutions of text-to-speech alignment and select one solution to be explored, implemented and tested. What mattered to us the most was to find a language independent solution that deals well with wide range of speakers and background noise. We have looked at each solution from several angles and found a solution to our liking (more about that solution in the next chapter 4). This chapter briefly touches on what sort of algorithms make each solution work as well as describes what advantages and disadvantages each solution brings to the table. What saddens us is the fact, it wasn't possible to test all the solutions presented in this chapter on the same data set, to compare them against each other and point out precisely how much better (or worse) each individual solution fares when compared with its peers. If there had been more time it might have been worth contacting all of the mentioned authors to ask them for their source-code in order to try and compare these aligners with one another.

3.1 Hidden Markov Models

The first approach we looked at for text-to-speech alignment is the use of Hidden Markov Models. The term Hidden Markov Models, describing the basic building blocks of this approach is so named after Andrey Markov, a genius scientist and mathematician who lived near the end of 19th century. Hidden Markov Models are probabilistic sequence models with very wide range of applications. Hidden Markov Models are used anywhere from weather predictions [17] to hand-writing recognition [16] and as such, it was only a question of time before someone considered using them for text-to-speech alignment. To better understand what a Hidden Markov Model is, it is important to first describe in detail what a Markov Chain is. It is best if we use an example to describe them, below 3.1 you can see a simple discrete-time Markov Chain with two states (one is Sunny and the other is Cloudy) describing current weather. On a sunny day, there is a 60% chance the weather will remain sunny (the arrow pointing from the sunny state back to the sunny state) and a 40% chance the weather will change to Cloudy (the arrow pointing out of the sunny state into the cloudy state). Please notice how the sum of outgoing probabilities (arrows) is always 1 (because that's how probability is defined).

Most real-world applications cannot just use a simple Markov chain because they often work with measured, inaccurate data and Markov Chains require all probabilities to be

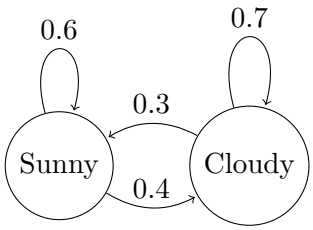


Figure 3.1: A simple Markov Chain

known. That is why Hidden Markov Models are used. For the purposes of modeling data using Hidden Markov Models, the data is divided into two categories: observed variables and hidden variables. Here 3.2 you can see a Hidden Markov Model depicted using Trellis Diagram. The bottom row represents the observed values while the top row represents hidden values.

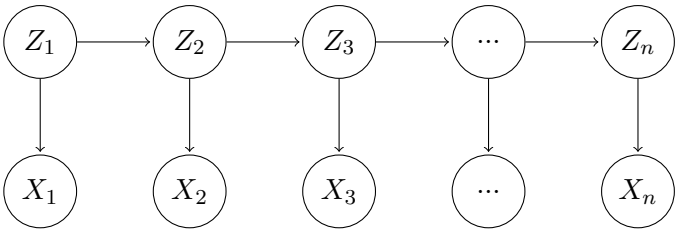


Figure 3.2: A Hidden Markov Model depicted using Trellis Diagram

3.1.1 Forward-Backward Algorithm

Forward-Backward Algorithm is used to calculate hidden state probability in Hidden Markov Models and it is built around the Markov Property: "given the present, the future does not depend on the past" [20]. This means we can take our model and break it on the current state Z_k into two parts: the past $\alpha_k(Z_k)$ and the future $\beta_k(Z_k)$ and treat each of these parts individually.

$$P(Z_k, X_{1..n}) = \alpha_k(Z_k) * \beta_k(Z_k)$$

The past is calculated using recursive Forward algorithm. The algorithm starts with Z_1 and continues up to Z_k

$$\alpha_1(Z_1) = P(Z_1) * P(X_1|Z_1)$$

$$\alpha_k(Z_k) = \sum_{Z_{k-1}=1}^{m} P(Z_k|Z_{k-1}) * P(X_k|Z_k) * \alpha_{k-1}(Z_{k-1})$$

The future is calculated using recursive Backward algorithm, it begins with the last element Z_n (as opposed to the first) and continues down to the Z_{k+1} element.

$$\beta_n(Z_n) = 1$$

$$\beta_k(Z_k) = \sum_{Z_{k+1}=1}^{m} P(Z_{k+1}|Z_k) * P(X_{k+1}|Z_{k+1}) * \beta_{k+1}(Z_{k+1})$$

This algorithm is sometimes called dynamic, because the previous calculations are not discarded and are simply reused when calculating next state. This results in shorter computation time.

3.1.2 Viterbi Algorithm

Viterbi Algorithm is named after Andrew Viterbi [30]. This algorithm is used to find the most likely sequence of hidden states in Hidden Markov Model. This sequence (or path) is then stored separately step by step as shown here 3.3.

$$\mu_k(Z_k) = max_{Z_{1:k-1}} P(Z_{1:k}|X_{1:k})$$

$$\mu_k(Z_k) = max_{Z_{k-1}} P(Z_k|Z_{k-1}) * P(X_k|Z_k) * \mu_{k-1}(Z_{k-1}) \quad for \quad k = 2..n$$

$$\mu_1(Z_1) = P(X_1) * P(X_1|Z_1)$$

Like Forward-Backward Algorithm, the Viterbi Algorithm is dynamic. As you can see, the

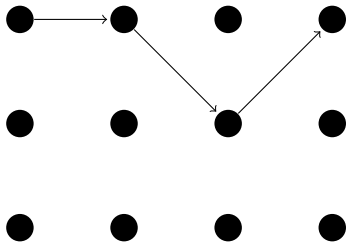


Figure 3.3: Calculated most likely path is stored step by step

algorithm reuses calculated values of μ. As an experiment, we try to use these values to predict the final F-score. The standard optimization of Viterbi algorithm is to only keep solutions that are within a certain distance of the best solution. This optimization is called beam search [15]. In our experiments, we observed 5.6 how with the increase of beam value more solutions are kept, and this leads to better alignment result.

3.1.3 Example Solution: Joint-sequence Model HMM Approach G2P Developed By FitVut

Our G2P solution was developed by Mikro Hannemean, Ph.D. and other members of our faculty. It is based on the findings of the following article [6]. It uses Viterbi alignment together with Joint-sequence models and is described in further detail in the following chapter 4. As we have later confirmed 5.5, our solution is both language and error independent, which makes it both unique and worth further examination.

3.1.4 Example Solution: JTrans

JTrans is a French text to speech aligner that is focused on user friendliness and ease of use. It uses forced alignment Viterbi, French Phonetizer and incremental block-Viterbi [9]. It is cited by this article [27] as being "state-of-the-art". Sadly, it only supports French at the moment and as such we were unable to test it without data. From the cited article [27] this solution appears to achieve precise results. JTrans is an open-source and freely available from github [2].

3.1.5 Segmentation Based Force Alignment

Normally, Hidden Markov Model Force-aligners are suitable only for short recordings. Sarah Hoffmann and Beat Pfister [15] have developed a G2P using segmentation-based approach: long speeches are first split into "sentence like" structures which are approximately aligned first and then, after further segmentations, aligned to an absolute word precision (using standard Hidden Markov Model segmentation methods) [15]. Segmentation based Force alignment is also language independent.

Table 3.1: Results of sentence alignment using unadapted models trained in three languages, taken from: [15]

	boundary errors		deviation (in ms)	
language	silent	non-silent	silent	non-silent
ger	4 (0.4%)	11 (34%)	1081	489
eng	1 (0.1%)	18 (35%)	354	193
fra	5 (0.5%)	7 (47%)	167	421
fin	1 (0.1%)	4 (19%)	261	76
bul	30 (3.5%)	4 (44%)	277	185
total	41 (0.9%)	44 (34%)	344	292

3.1.6 Example Solution Using Subspace Gaussian Mixture Models

In most Hidden Markov Models solutions, a relatively large number of parameters represent a single HMM state (see diagram above 3.2). In Subspace Gaussian Mixture Models all of the parameters are constrained to live in a relatively low dimensional subspace that is common to all the states. This allows the model to be trained on all languages simultaneously. Conclusively, this leads to a language independent solution. A group of researchers from all over the world, including members of our faculty, have developed and tested such a solution in this article [8], below is a table taken from their article.

Table 3.2: Amounts of data for acoustic model training and testing, taken from [8]

	Shared parameters trained on	Word Error Rate [%]
Baseline	n/a	70.5
Subspace Gaussian Mixture Models	1h English	67.6
	Spanish + German	59.8
	+ 1h English	59.6

3.2 Landmark Phoneme Recognition

The fictional principle is very similar to solutions using Hidden Markov Models, but unlike them the alignment takes place directly in the text domain. The audio is first segmented into phonemes, which are then aligned to graphemes with a dynamic programming edit-distance transformation [14]. Simply put, the text is aligned based on well-known graphones.

3.2.1 Example Solution

Alexander Haubold and John R. Kender [14] have successfully came up with this method. While the quality of aliment wasn't very high (see table below) their solution was language independent.

Table 3.3: Speech-text alignment stats and results, taken from [14]

	Avg. Matching Error	# Speech phonemes	# Text phonemes	Copies	Deletions	Insertions	Replacements
A	3.9	64265	27645	16695	39189	2569	8381
B	7.7	64265	21608	14121	43997	1340	6147
C	6.43	54537	16248	11459	38947	658	4131
D	26.73	12596	4520	2960	8395	319	1241

3.3 Speech Synthesis Based Approach

A speech Synthesis based alignment uses a speech synthesizer to first create a speech pattern and then align speech based on this pattern. [19]. In order to successfully compare the generated synthetic speech and the original speech multiple characteristics must be extracted [19]:

- First C_i cepstral coefficients from linear prediction analysis
- Δ_{C_i} Delta of cepsal coefficients
- E Normalized energy of each frame
- ΔE Delta of normalized energy

Afterward, segmentation based on "Dynamic Time Warping" algorithm takes place [19]. This algorithm tries to minimize the accumulated distance between generated a and original b frames of speech signal.

$$d(a,b)^2 = \alpha \sum_{j=0}^{N} (C_j(a) - C_j(b)))^2 + \beta \sum_{j=0}^{N} (\Delta_{C_j}(a) - \Delta_{C_j}(b))^2$$

$$+\gamma(E(a) - E(b))^2 + \phi(\Delta E(a) - \Delta E(b))^2$$

The letters from Greek alphabet stand for general weighting coefficients.

$$\alpha = 1.0, \beta = 1.25, \gamma = 1.25, \phi = 1.25$$

The great advantage of Speech synthesis-based alignment is that there is no training stage and no training database is required [19]. The biggest drawback is speaker dependency of the system. We chose against this method, because we use a very diverse data set from a wide range of speakers.

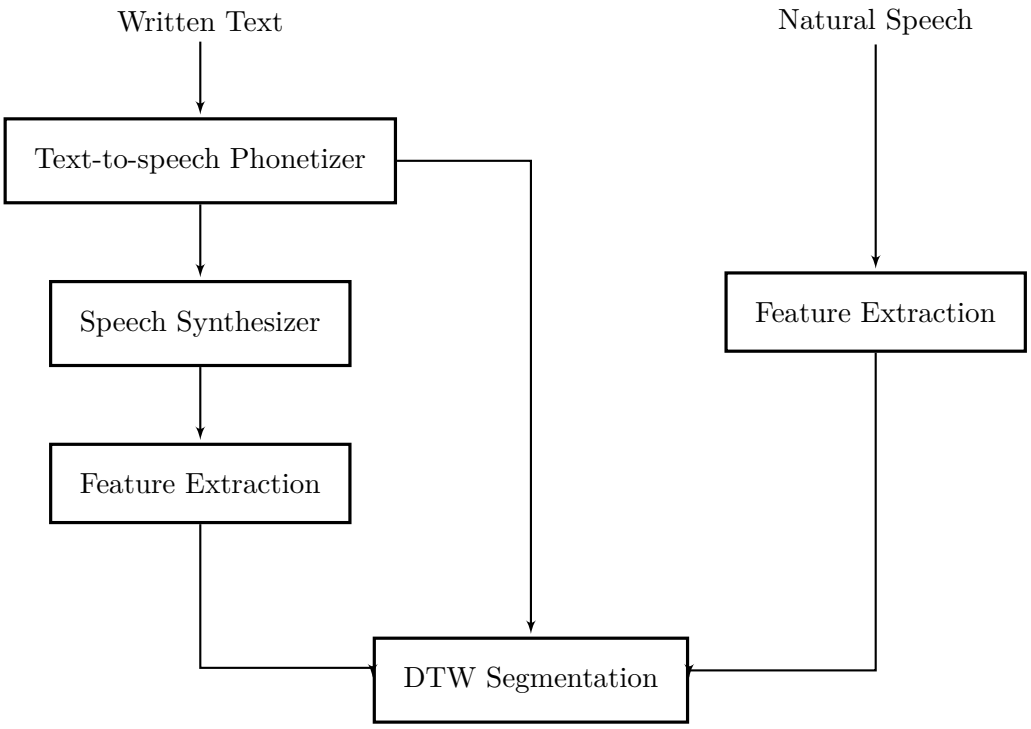


Figure 3.4: Phonetic Segmentation Process, taken from [18]

3.3.1 Example Solution: Fabrice Malfrère And Thierry Dutoit

In the previously cited article [18] Fabrice Malfrère and Thierry Dutoit experimented with high quality speech synthesis in the context of automatic phonetic segmentation and received very respectable results. Their system was partially speaker independent (except for speaker sex dependency). Their system was also language dependent and thus unusable for any languages without existing high-quality speech synthesizers.

3.4 Bayesian Based Approach

Named so after Thomas Bayes [11], Bayesian models are all based on the Bayes theorem (an interesting fact is, it wasn't Thomas Bayes who came up with this theorem, it was Laplace years later after Thomas Bayes' death [11]):

$$P(B_t|A) = \frac{P(A|B_t)P(B_t))}{\sum_j P(A|B_j)P(B_j))}$$

3.4.1 Pitman-Yor Process

Pitman-Yor processes are nonparametric Bayesian models [29] A Pitman-Yor process G is described using this formula:

$$G \sim PY(\alpha, d, G_0)$$

Where α stands for base rate, d stands for discount parameter and G_0 stands for mean distribution. The following must always be true:

$$0 \leq d < 1$$

$$\alpha > -d$$

To better understand this process, imagine a restaurant with an infinite number of tables [10]. The first N customers walk in and are seated at their K tables. Afterwards, the Nth customer arrives. The probability that this customer will sit at a new table is proportional to $\alpha + Kd$ and she sits at a table previously occupied by someone with probability proportional to $\#_k - d$, where $\#_k$ stands for the number of customer already seated at a table k. With each of these tables we associated a parameter Θ_k that comes directly from G_0 [10]. This selection is exchangeable: the distribution does not care about customer order [10].

3.4.2 Example Solution: Mirko Hannemann

A member of our faculty, Mirko Hannemann Ph.D., has successfully implemented Bayesian Hierarchical Pitman-Yor Language Model [13] and has reached very respectable results. These results were the reason why this work wanted to use the HPYLM solution instead of the Joint-Sequence Model Viterbi Hidden Markov Model it is using now. The main problem with this experimental solution is that is hasn't been parallelized yet and is therefore unusable with longer recordings (we don't have enough physical memory in our server). Nevertheless, an optimized, parallelized solution has been promised and is currently in development as of writing of this work. In the meantime, anyone can download an early version of this solution from GitHub [1]. We strongly recommend any future readers to do their own experiments with this solution. Below 3.4 are the results Mirko Hannemann Ph.D. has achieved.

	phoneme error rate	word error rate
7-gram JSM trained with Sequitur	5.92%	24.65%
Bayesian HPYLM G2P after three iterations	5.92%	24.73%
averaging several HPYLM (Phonetisaurus [22])	5.80%	24.36%

Table 3.4: Mikro Hanneman's Pitman-Yor solution performance, from [13]

3.5 Neural Network Based Solutions

Artificial neural networks are computing systems that are able to "learn" almost anything and as such it should theoretically be possible to use them for text-to-speech alignment. While Neural Networks aren't very commonly used for text-to-speech alignment they hide a tremendous potential for the not-so-far-away future. Neural networks are composed from one to many neurons such as the one pictured here 3.5. The biggest advantage of using neural networks is their ability to learn and improve over time with use. Their disadvantages include the necessity of obtaining a large amount of both positive and negative samples (this has caused us complications when we were attempting to train a neural network in chapter 6) before the alignment itself can take place. This means, it is impossible to train neural network on data which is to be aligned. Furthermore, a neural network is likely to perform better on data on which it has been trained or data similar to the data on which it has been trained.

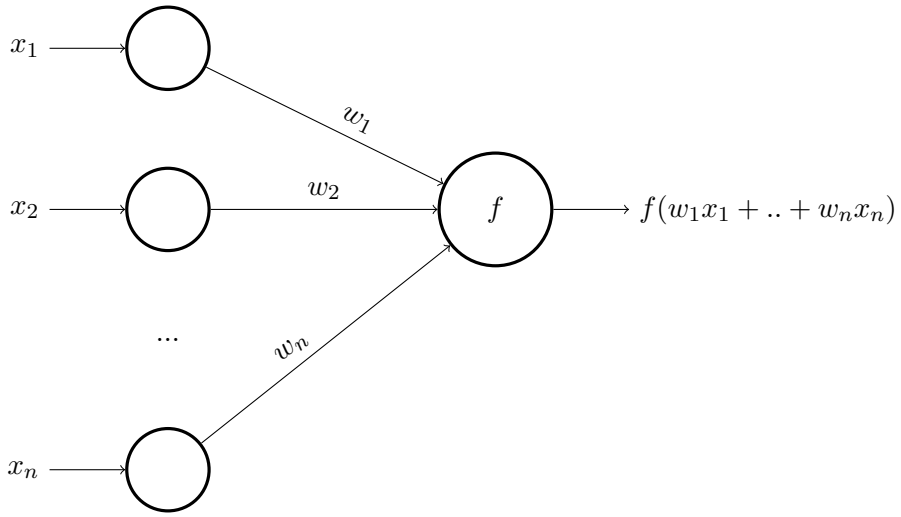


Figure 3.5: Weighted Input Summation Neuron , taken from [21]

3.5.1 Example Solution: Guillaume Serriere

A group led by Guillaume Serriere [27] have developed a text-to-speech alignment Deep Neural Network (DNN). To train it they used the output from ASTALI and JTrans existing aligners to produce positive examples. Below is a table comparing their results with above-mentioned JTrans.

	Acoustic baseline	Neural Network	JTrans / ASTALI
Equal Error Rate (EER)	48%	36%	
Precision (median)	43%	52%	60%
Recall (median)	53%	69%	45%
F1 (median)	48%	60%	51%

Table 3.5: Guillaume Serriere's Neural Network compared to JTrans, from [27]

Chapter 4

Our G2P Alignment Solution

Our G2P uses scripts first developed by a member of this faculty – Mirko Hanneman, Ph.D. The G2P works with the findings of this article [6] and it uses the Viterbi algorithm together with Joint-Sequence Models. We have selected this G2P because due to its smart Joint-Sequence Model building and clever pruning, it is able to align recordings of unlimited length (the only limitation is the amount of physical memory available). As requested, it can align recordings of any language (so far Russian, Hungarian, Czech and English have been tested). Contrary to many other algorithms that require training, our G2P can be trained on the same data set as the data set to be aligned. What's more, it is (to a limited degree) error-independent, and it allows the user to decide between alignment precision and computation time 5.6.

4.1 Joint-Sequence Models

Our work heavily centers around Joint-Sequence Models [6]. As the name suggests, Joint-Sequence Models model the joining of sequences of graphemes to sequences of phonemes. This joining, as noted by following article [13], is almost never isomorphic due to the great variance of most organic human languages (exceptions are artificial languages like Esperanto). In Joint-Sequence Models all possible graphone sequences are first generated, then their individual probabilities are calculated and finally the most likely solution is found.

4.2 Using Our Solution

It is important to describe how to use our solution, should any of you – future researchers want to use it. First, you take your audio (.wav) files and convert them to sets of phonemes (.rec) using any phoneme recognizer. We chose PhnRec for this task, because it was developed by members of this Faculty and as such it was possible to communicate with its developers and compare a well-trained phoneme recognizer with a less trained phoneme recognizer. Next, you extract text files from XML (or the reference file, more about that later) using our scripts: getlabs.sh for standard XML and getsubtitles.sh for reference subtitles (these scripts function as a handy "how to" guide on subtitle extraction). Once you have both phoneme files (.rec) and grapheme files (.lab) you used the prapare_labels.sh script which merged these two into one (.G2P_labels). These files are then used by the G2P aligner itself (G2P_alignment.py). Our G2P aligner outputs three types of files: a

file containing graphones (.graphones), a file containing Viterbi costs (.costs) and a file containing alignment (.alignment). To simplify things, we made our G2P output Vitebi costs into the .alignment file as well and as such we do not need to use .graphones or .costs files. The G2P output does not contain any formatting and as such is cannot be simply opened and read by human beings. This is what alignment2mlf.py script is for, it takes the G2P output and merges it together with phoneme file into a formatted output. From this output we derive all information, whether it is duration of individual words (using awk) or Viterbi probability values used by our neural network 6. Our solution is built to run on 64-bit Linux and requires a sizable computer memory or swap drive (the memory required is depended on the length of your recording).

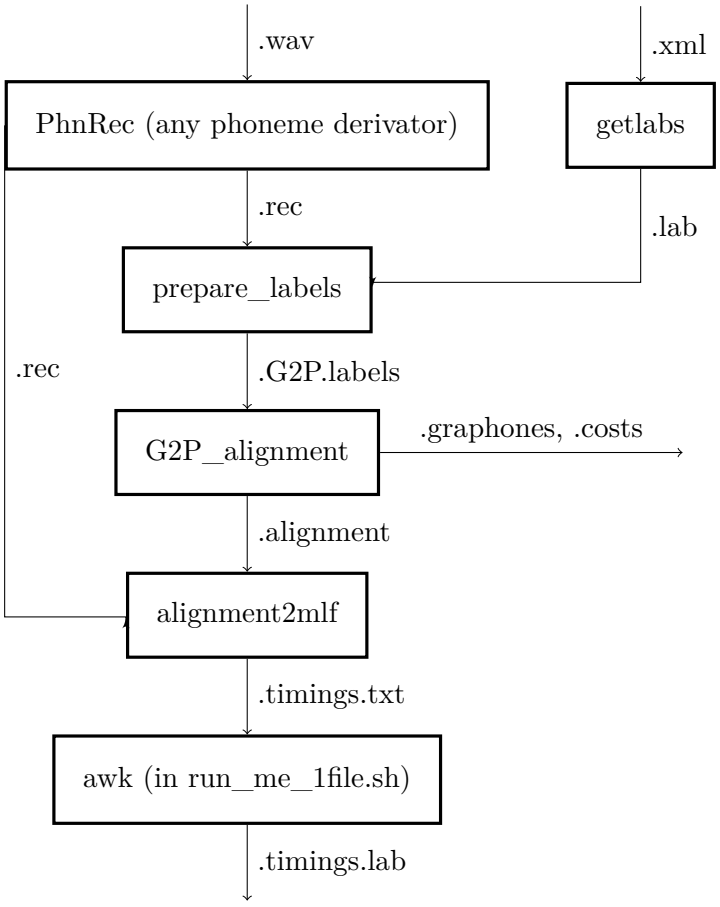


Figure 4.1: Diagram of our solution

4.3 Phoneme Derivation: PhnRec

To derive phonemes out of audio files (.wav – Waveform Audio File Format) this work has chosen to use PhnRec simply because this software tool was developed by members of this faculty. At its core, PhnRec uses hybrid Hidden Markov Model / Artificial Neural Network system.

000000	1800000	pau	-37.213451
1800000	2200000	ih	-11.939968
2200000	2500000	dx	-7.539501

Table 4.1: An example of PhnRec phoneme output

PhnRec output format is shown in the table 4.1: the first two columns describe the time duration of each phoneme (where 10000000 is equal to one second), phonemes themselves are in the third column, then the fourth column describes the probability that each phoneme has been identified correctly. Before any version of PhnRec can be used all lines containing the "silent" phonemes *sil* must be removed, as this was found to reduce the quality our alignment.

4.3.1 Phnrec v2_21 (titled OLD)

This, publicly available (from this website [3]), version of PhnRec is based on the findings of Petr Schwarz, Ph.D. as stated in his Ph.D thesis [24]. This version of PhnRec has been trained on relative low amounts of data (1.71 hours for English and 9.72 hours for Czech [24]) which explains its relatively disappointing results 5.1. The publicly available version currently supports four languages: English, Czech, Hungarian and Russian. The following languages have been tested but were not made available to download: German, Hindi, Japanese, Mandarin and Spanish.

4.3.2 Better Trained Version Of Phnrec (titled NEW)

There is also a newer, much better trained version which is used in chapter 5, this version hasn't been given a number yet and is therefore referred to as NEW (and the v2_21 is referred to as OLD). This new version is based on the findings of the original conference paper [25] but this version uses split context Artificial Neural Network [26] features with bottle-neck inside the ANN [12] as the final phoneme classifier. The output is 3-state phoneme posteriors, which is afterward sent to the decoder. The New PhnRec Czech version was trained on 300 hours and the English version was trained on 2000 hours. Both datasets included samples recorded using proximate and distant microphones with and without artificially induced noise-pollution.

4.4 Subtitle Set: reference (ref.)

The, so called, reference subtitles were taken from the dev.full.ref.ctm reference document. Their purpose was to serve as a reference when evaluating the quality of our alignment. That way we could find what our G2P can do when it is not burdened with grapheme errors. It was later revealed that our Joint-Sequence Hidden Markov Model prefers human-like formatting of transcript_human even while being compared with the reference document (more on that see 5.3). This has caused reference not to have the best results contrary to all expectations. The text itself was computer generated and as such carries a certain degree of imperfection (for example: $BBC = B.B.C.$).

4.5 The Three Main Subtile Sets

All these subtitles were extracted from the XML files provided for the Multi-Genre Broadcast Challenge. Each of these files consists of three or more different tracks of subtitles. The first is transcript_orig is described by the XML files as "output of the parsing of the original transcription files provided by the BBC". The second is transcript_align is described as "forced-alignment of the original transcription". The third is transcript_human, described as "manual transcript based on force-aligned subtitles provided" which has shown some unexpected results (more on that see 5.3). There was also transcript_lsdecode, described as "output of the lightly supervised decoding", but since it wasn't present in all XML files and as such it wasn't tested.

After selecting the correct nodes from the XML tree (only selecting segments with the correct annotation_id value), they are printed in this format "0 0 WORD 0" into a .lab file. This formatting is required by the G2P.

4.6 Processing Output

Our G2P prints output into a .timings.txt file which is then converted into .timings.lab (see the figure 4.1 for more information) in a format similar to PhnRec: three columns, first two describe duration of a word (yet again, 10000000 is equal to one second), the third is the word itself.

0	2800000	HELLO
2800000	4400000	AND
4400000	14100000	WELCOME

Table 4.2: An example of G2P output

4.7 Scoring

We score our results using scripts developed by Thomas Hain for Multi-Genre Broadcast Challenge 2015. These evaluation scripts were developed as part of the EPSRC programme in Natural Speech technology. It calculates Precision, Recall and F-score for each individual file and also for all files together. When evaluating individual test-case we focus on the overall F-Score as this helps us to see the bigger picture rather than to burden ourselves with insignificant anomalies. The only time when we write down individual files is when we encounter an anomaly and want to bring the readers attention to this anomaly.

4.7.1 The Scoring Process

In order to convert the output into the a format useful for evaluation we take information from all .timings.lab files and change it thus: we make the first column to be the base name of the file we are currently processing (without any extensions), the second column (separated by a space character) be a '0' character, the third column is word's beginning time divided by 10000000, the fourth column is the delta (difference) between the beginning and the end of current word divided by the same value.

After we have successfully converted our files into a form shown in the table 4.3, we merge them all into a single .ctm file. This singular file is then the only thing that is being

20080505_180000_bbcfour_the_book_quiz	0	0.00	0.28	HELLO
20080505_180000_bbcfour_the_book_quiz	0	0.28	0.16	AND
20080505_180000_bbcfour_the_book_quiz	0	0.44	0.97	WELCOME

Table 4.3: An example of G2P in an output suitable for scoring

evaluated by these evaluation scripts. After encountering difficulties with having special symbols inside the .ctm files, any line containing a potentially dangerous symbol is removed prior to evaluation ('$', '[', ']', ')', '(') as well as any line not containing a capital letter (subtitles are always composed of capital letters).This way we avoid crashes and failures these characters tend to cause when evaluating.

Chapter 5

Experiments, Problems And Solutions

In this chapter we experiment with, and explore our G2P solution. Looking at it from many different angels, we try to find and explain its strengths and weaknesses. We delve deep into what influences our results the most. In addition, we attempt to strike the best compromise between accuracy and CPU time. This chapter showcases only some of our results from our G2P alignment experimentation. As requested, this chapter also attempts to describe, to the best of its ability, what causes a failure during the alignment process.

5.1 The Influence Of Phoneme Derivation Accuracy

A set of reference subtitles has been aligned with phonemes generated using a less trained phoneme derivation algorithm and phonemes generated with "well trained" algorithm. This test was aimed at determining what level of influence, if any, phoneme derivation accuracy has.

This test was done using:

- text: reference
- phonemes: OLD PhnRec EN, NEW PhnRec EN
- beam: 300

Bound	OLD PhnRec EN	NEW PhnRec EN	% Improvement
100	0.0070	0.4540	6385.7143
300	0.0231	0.5753	2390.4762
1000	0.0639	0.6714	950.70423
3000	0.1299	0.7295	461.58584

Table 5.1: The influence of phoneme derivation accuracy on F-score

Interestingly, there is a great magnitude of difference between these two columns, this suggests that to improve text-to-speech alignment we should first and foremost focus on improving phoneme derivation accuracy. Below is a chart that tries to showcase just how much precise phoneme derivation can help alignment accuracy. As you can see from the

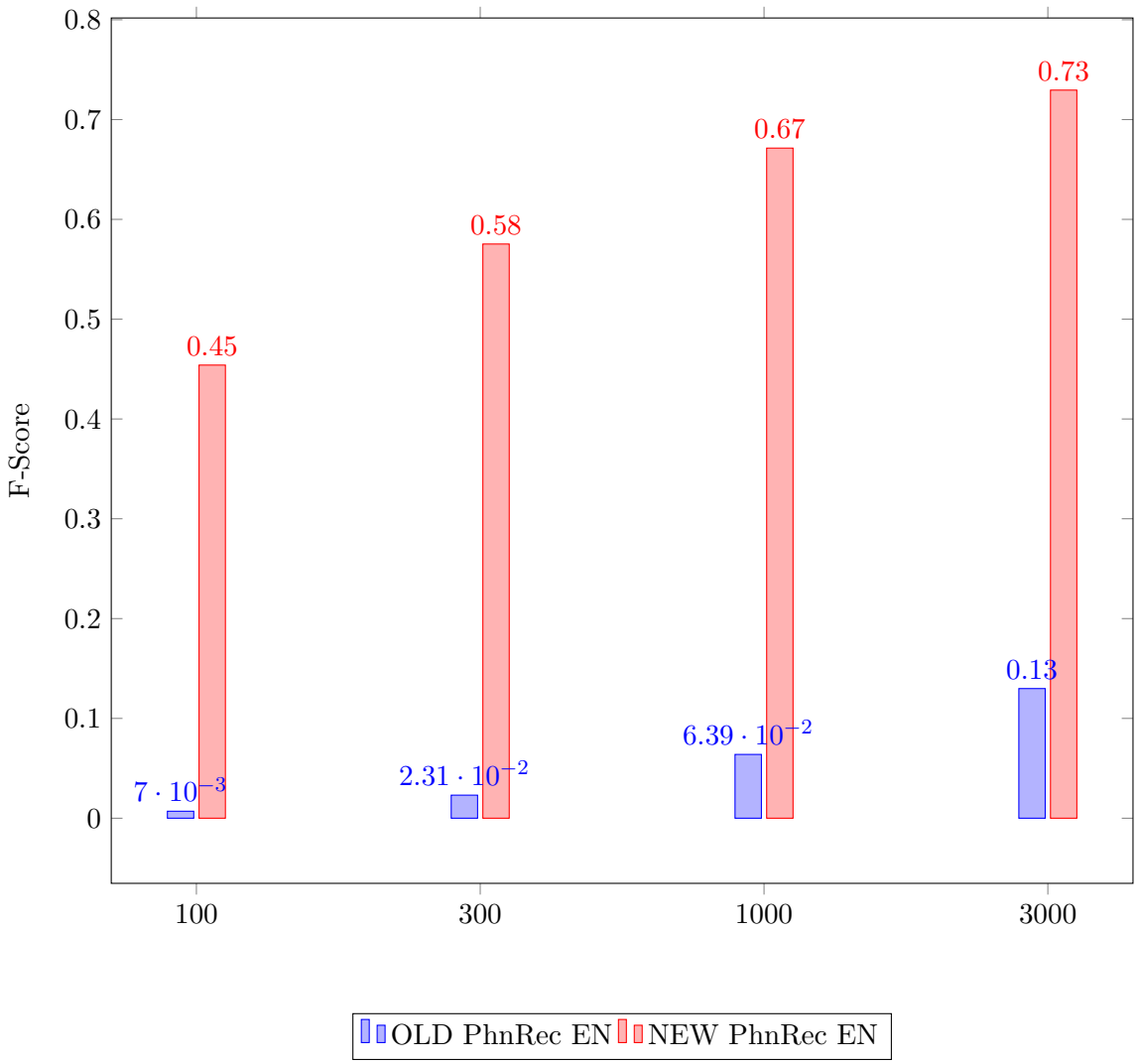


Figure 5.1: The magnitude of influence of phoneme derivation accuracy on F-score

chart, the difference between these two columns is truly enormous. We can therefore deduce our G2P does not deal well with phoneme inaccuracies.

5.2 Inaccurate Sets Of Subtitles

This series of tests serves to measure what happens when the subtitles carry a certain degree of inaccuracy. We have selected four tracks of subtitles: the reference, transcript_orig, transcript_align, and transcript_human to compare against each other.

This test was done using:

- text: reference, transcript_orig, transcript_align, transcript_human
- phonemes: NEW PhnRex EN
- beam: 300

Bound	reference	transcript_human	transcript_align	transcript_orig
100	0.4540	0.4778	0.1897	0.1746
300	0.5753	0.5878	0.2510	0.2330
1000	0.6714	0.6699	0.3143	0.2946
3000	0.7295	0.7169	0.3668	0.3472

Table 5.2: Results of alignment using different sets of subtitles

We can observe something very unexpected has happened here. Transcript_human seems to have outperformed the reference text, the same text that it was evaluated against. This seems to be only the case when we evaluate with less than 300ms tolerance (boundary). We have decided this theory deserved further examination.

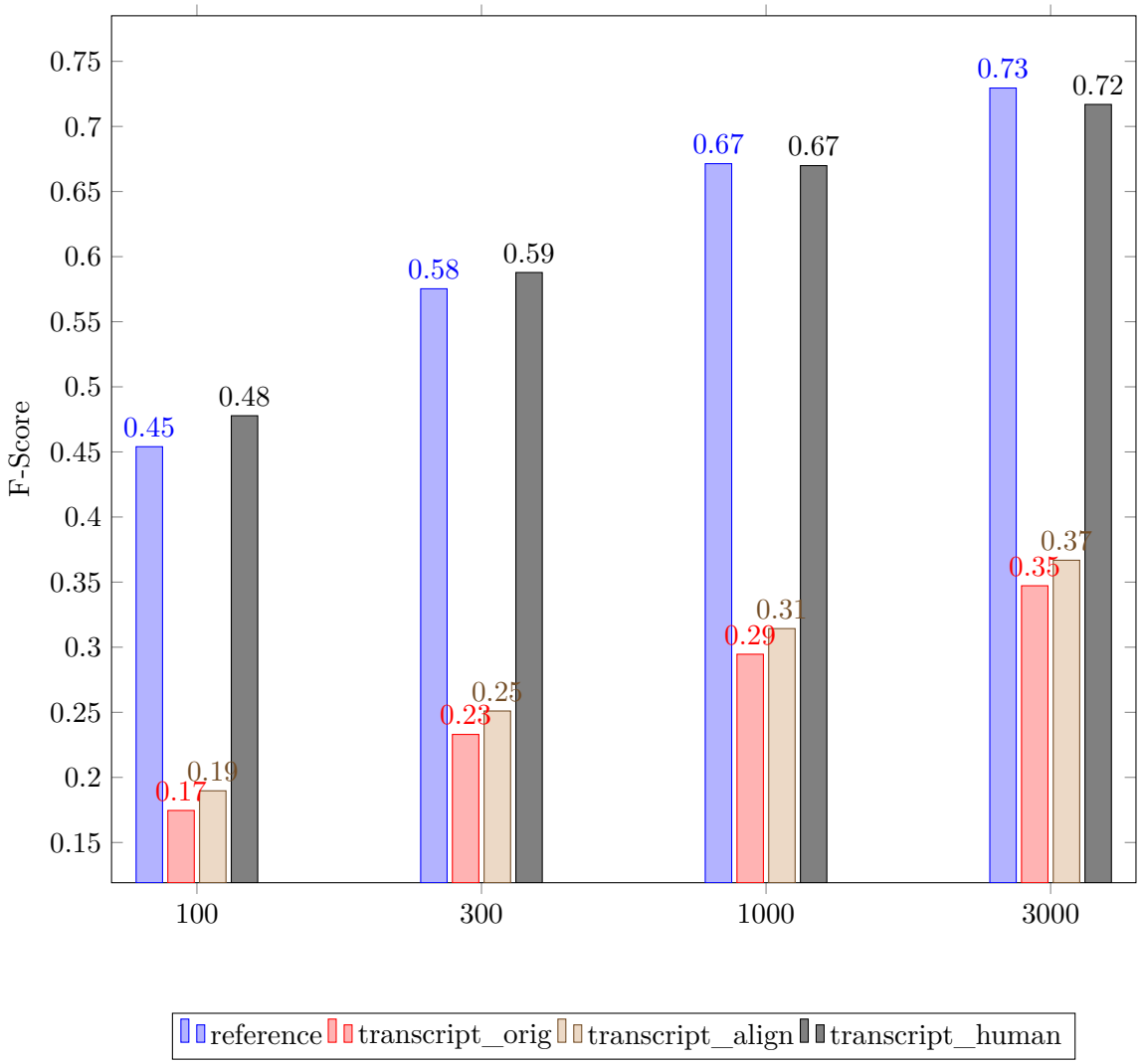


Figure 5.2: The magnitude of influence of phoneme derivation accuracy on F-score

5.3 Further Examination Of The Anomaly

In this test scenario, we wanted to eliminate the possibility that this anomaly was a "one off" occurrence and truly confirm whether better formatted incorrect subtitles outperform their reference counterpart. In this section our Hidden Markov Model was fed incorrect data on purpose, in order to replicate and validate the unexpected results we had achieved in the previous experiment.

This test was done using:

- text: reference, transcript_human
- phonemes: NEW PhnRex CZ
- beam: 300

Bound	reference	transcript_human
100	0.2556	0.2765
300	0.3398	0.3618
1000	0.4131	0.4327
3000	0.4735	0.4841

Table 5.3: Different language comparison

Unexpectedly, transcript_human outperforms the reference subtitle set, this confirms the theory that our Hidden Markov Model performs better when fed incorrect graphemes. Since this is only the case with transcript_human and not with transcript_orig and transcript_align, I propose this is purely due to better word formatting (transcript_human was written by a person, whereas reference was generated).

5.4 Grapheme Error Examination

In 5.1 we have observed that the relative F-score improvement seems to decrease rapidly with the increase of boundary. In this test scenario, we tried to examine whether this is also the case with grapheme error.

This test was done using:

- text: reference, transcript_orig, transcript_align, transcript_human
- phonemes: NEW PhnRex EN
- beam: 300

Bound	reference	transcript_align	% Improvement
100	0.4540	0.1897	139.32525
300	0.5753	0.2510	129.20319
1000	0.6714	0.3143	113.61756
3000	0.7295	0.3668	98.882225

Table 5.4: Boundary and % improvement relation when the error is purely grapheme based

We have discovered that when the grapheme error is purely grapheme based the relative improvement with the increase of boundary is much slower than when the error is phoneme based.

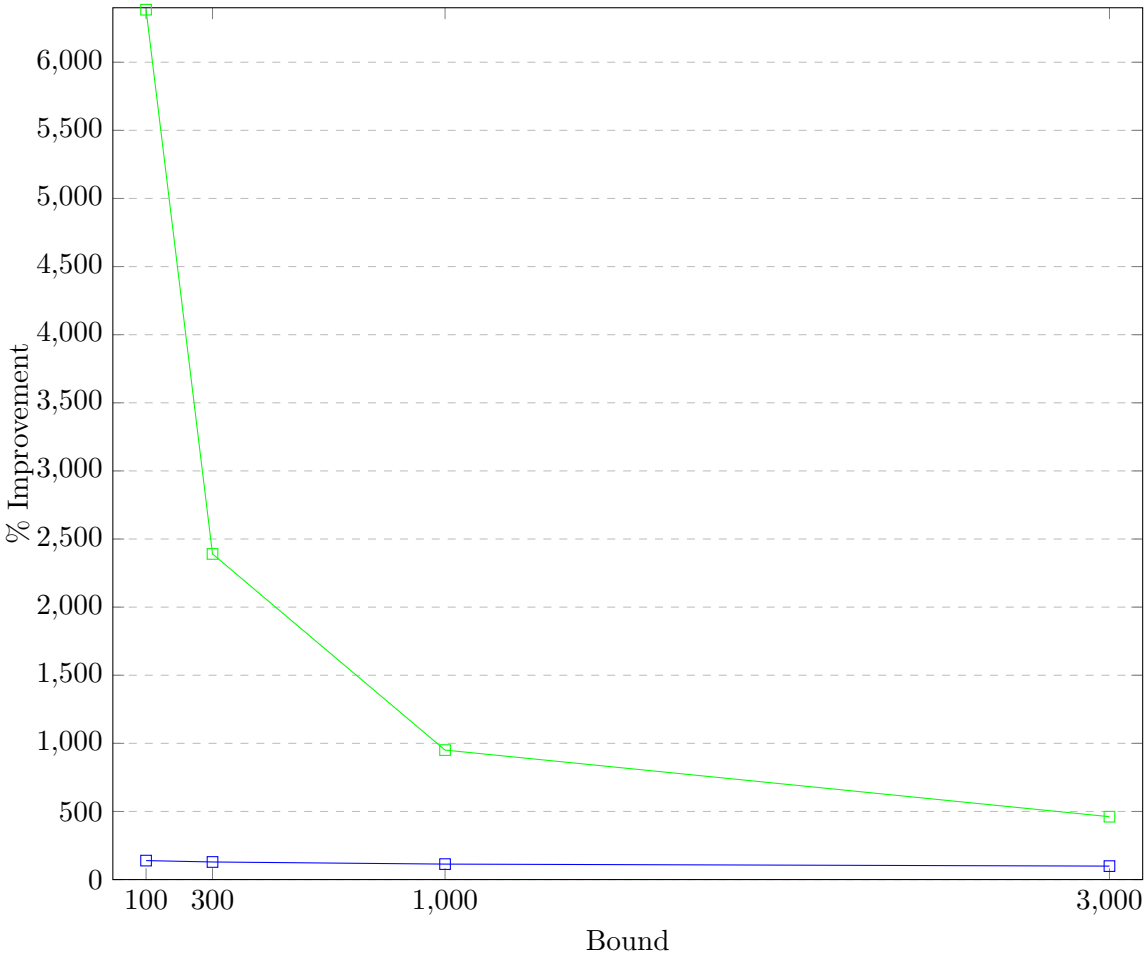


Figure 5.3: % Improvement comparison between grapheme and phoneme-based errors

5.5 Language Independent G2P Alignment

Finding language independent audio-alignment was one of the tasks we have set out at the beginning. Here we tested whether our Joint-Sequence Model Viterbi is truly such a solution.

This test was done using:

- text: reference
- phonemes: OLD PhnRec CZ, NEW PhnRex CZ, OLD PhnRec EN, NEW PhnRec EN
- beam: 300

As you can see above, a well-trained G2P algorithm will outperform a less trained algorithm regardless of the language used. This suggests that language neutral text-to-speech

Bound	OLD PhnRec CZ	NEW PhnRec CZ	OLD PhnRec EN	NEW PhnRec EN
100	0.0070	0.2556	0.0070	0.4540
300	0.0231	0.3398	0.0231	0.5753
1000	0.0639	0.4131	0.0639	0.6714
3000	0.1299	0.4735	0.1299	0.7295

Table 5.5: Different language comparison

alignment is indeed possible and should be explored with further research.

It is important to note that the following two audio track alignments have failed (in case of OLD PhnRec CZ). We assume, incorrect phoneme derivation increases the size of graphone matrix which leads to MemoryError and subsequent crash.

- 20080511_113000_bbctwo_premiership_rugby
- 20080510_174500_bbcone_doctor_who

The imperfect phoneme derivation projects strongly in the final F-score (as we have established 5.1) and produces the result F-score of 0. A great number of tracks have failed this way when using less-trained Russian or Hungarian versions of PhnRec and this further confirms our theory.

5.6 Beam Width Influence

In this scanarion, we examined the relation between beam cut-off value and the final F-score. We measured this relation with four different boundary values and compared our results in order to find the best beam value to evaluate with. As higher beam value results in higher computation time it is crucial to strike a balance between alignment precision and computation time.

This test was done using:

- text: reference
- phonemes: NEW PhnRec EN
- beam: 10, 30, 50, 100, 150, 200, 250, 300, 350, 400

Bound	Beam: 10	30	50	100	150
100	0.0363	0.1215	0.1806	0.2947	0.3651
300	0.0598	0.1615	0.2331	0.3699	0.4529
1000	0.1042	0.2101	0.2879	0.4372	0.5254
3000	0.1929	0.2682	0.3429	0.4888	0.5741

Table 5.6: Beam width influence part 1

Bound	Beam: 200	250	300	350	400
100	0.4164	0.4314	0.4540	0.4659	0.4784
300	0.5183	0.5393	0.5753	0.5922	0.6074
1000	0.6009	0.6288	0.6714	0.6950	0.7090
3000	0.6518	0.6854	0.7295	0.7549	0.7667

Table 5.7: Beam width influence part 2

As you can see from the two tables 5.6 and 5.7, with the increase of beam value the F-Score seems to increase. This increase seems to slow down the higher the beam value goes. From this data, we suggest using a beam value between 200 and 300 as anything more than that results in unnecessary extra computation time with negligible increase in F-score.

When the data from the table 5.6 was taken and plotted out in 5.4 there was one thing that immediately stood out to us: all the curves appeared logarithmic. After further examination using Matlab this fact was confirmed. Resulting logarithmic functions are stored in the following table 5.8.

Bound	Logaritmic Function
100	0.130471 log(0.102916 x)
300	0.161015 log(0.109911 x)
1000	0.178253 log(0.13154 x)
3000	0.172374 log(0.202766 x)

Table 5.8: Beam logaritmic functions

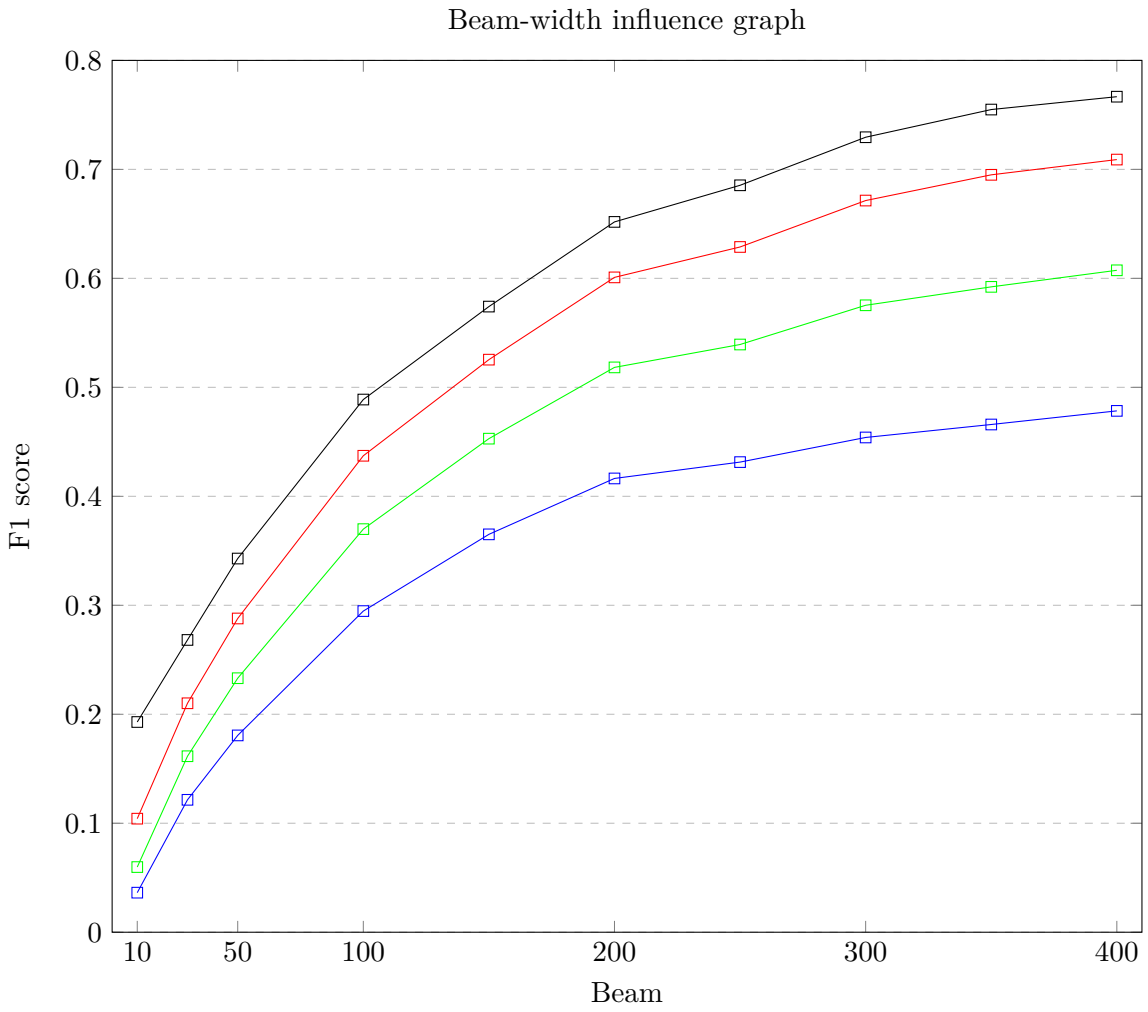


Figure 5.4: Relation between F1 and Beam: blue = 100ms boundary, green = 300ms boundary, red = 1000ms boundary, black = 3000ms boundary

One thing we can deduct from graph 5.4 is that there is not much point in increasing the boundary value for evaluation when the beam limit has been set very low. The F-Score improvement from using a higher boundary seems to increase with a higher beam score.

Chapter 6

G2P Alignment Score Prediction

In this chapter, we explore whether it is possible to approximate the accuracy of G2P alignment (predicting deletions) given the outputs of our Viterbi algorithm: G2P logarithmic likelihood, current beam value, and accumulated logarithmic likelihood using neural networks. We explore whether a neural network is even suitable for such a task.

6.1 Data Preparation

We run our scoring script with the ' - - matchres' option which returns one huge file with all words, each of the words evaluated. = signifies a correct alignment, I signifies an insertion of an unnecessary word, finally D signifies the deletion of a word that should have been aligned to that time but wasn't. Deletions are of special significance to us, as we are trying to predict their occurrences even without the reference document. Our data is then synchronized by time with the data from .timings.txt and stored in one of more .csv files to be used as an input for out neural network. Each line stands for 10 milliseconds.

Listing 6.1: Neural Network input file example (CSV)

```
G2P loglik, beam, ac loglik, error
...
7.169593216566667, 4.47440600679, -4.301755929899997, 0
8.127403532672727, 4.70668906641, -4.3377929128999995, 0
8.127403532672727, 4.70668906641, -4.3377929128999995, 0
...
```

For our Neural Network we are using Matlab R2017 with the following parameters:

6.2 The Treatment Of Silence

We have copious amounts of data which is the ideal situation when training a neural network. However, our data is not defined on the whole track, instead our values are only defined from the beginning to the end of each phoneme. To get around this problem, it was important to decide how to treat these undefined time periods "in between". The first and most obvious approach was to simply keep using the values taken from the previous phoneme. The second approach was not to include (delete) all silent periods. The option, we have come up with was to write $0, 0, 0, 0$ repeatedly during silence. As you can see from the second column in table 6.2, our results were completely disappointing, we didn't

Type	Pattern recognition and classification (nprtool)
Input	All columns in csv except for last
Output	The last column
Data division percentages	70% Training, 15% Validation, 15% Testing
Number of Hidden Neurons	10
Data division	Random
Training	Scaled Conjugate Gradient
Performance	Cross-Entropy
Calculations	MEX
Trained on	20080509_180000_bbcfour_world_news_today
Tested on	20080511_172500_bbcone_bbc_london_news

Table 6.1: Neural Network setup

Table 6.2: Different approaches to the tratment of silence

	% Error Rate	% Correctly detected 1
Skipping silence	23.90	0
Propagating old value	19.86	0
Replacing silence with zeros	19.90	0
"0 wire" a control script	19.86	0

manage to detect a single deletion (denoted as 1 in our data). Our theory is that it is due to the disproportionate surplus of negative samples (lack of positive samples). As you can see from the table there are only about 19% of ones (deletions) in our training data sets and that causes a control script (0 wire which always outputs zero, meaning no error detected) to perform about as accurately as our trained neural networks.

6.3 The Treatment Of Silence In Context

An interesting theory was proposed: the inclusion of context. Each line had previous 10 lines included to allow the neural network to look both into the "future" and the "past". As

Table 6.3: Different approaches to the tratment of silence with the context of 10

	% Error Rate	% Correctly detected 1
Skipping silence	23.80	0.1
Propagating old value	20.1	0.8
Replacing silence with zeros	18.8	9.9
"0 wire" a control script	19.87	0

you can see in the table 6.3, the inclusion of context seems to have improved the situation a bit. But with only 9,9 percent of deletions successfully detected, we began to ask ourselves, whether the detection failure is not due to it being trained only on one recording.

6.4 G2P Score Prediction: Conclusion

As far as we discovered, it is difficult to design a neural network capable of predicting alignment F-score. This could be for several reasons, it might be simply because the overall length of deletions is much less than the rest of the recording and as such there is not enough positive samples to train a neural network correctly. It can be that neural networks require much larger context that what we have been using (the greater the context the longer the training process takes time). What we have later discovered is there is an easy analytical solution to this problem, read chapter 7 to know more.

Chapter 7

Accidental Analytical Solution To G2P Score Prediction

After experimenting with neural networks over and over, we nearly gave up hope that such a thing as alignment accuracy prediction could even work with our data. When dealing with a different sort of experiment in 5 we were trying to examine what has caused some recordings to under-perform to such a large degree, partially out of curiosity and partially out of hope of finding our problem we have decided to plot out the beam values and Viterbi outputs of all of the misbehaving tracks. This, at first seemingly futile activity gave birth to a new a theory that could help us with our G2P score prediction problem.

7.1 Plotting

We have devised a special script (plot_timings_mod.py) to help us plot out all relevant information. This simple script allows us to see deletions (red interrupted line on top), correctly aligned words (blue interrupted line on top), and it allows us to see the G2P logarithmic likelihood (green in the middle), current beam value (red line), and accumulated logarithmic likelihood (blue in the middle). In the current state plot result images are stored in the png format which leads to unnecessary large files (2.8MB multiplied by number of files multiplied by number of test) As a foresight, a .jpg compressed format should have been used.

7.2 Observation One: Lower Beam Is Good

First, we plotted out result of evaluation of 20080509_180000_bbcone_the_one_show, which at 300 beam reached a respectable F-score of 0.7672 for transcript_human and not so respectable F-score of 0.2274 for reference (100ms boundary). We did by no means expect such a perplexing result, so we were keen to discover what had caused it. Below you can see the result of our plots.

There wasn't much to see in both pictures, except for the fact that the better performing track had the beam somewhat higher than the less performing track. Is this what we have been looking for with our neural network? We decided to delve deeper into the topic.

Figure 7.1: Reference 20080509_180000_bbcone_the_one_show

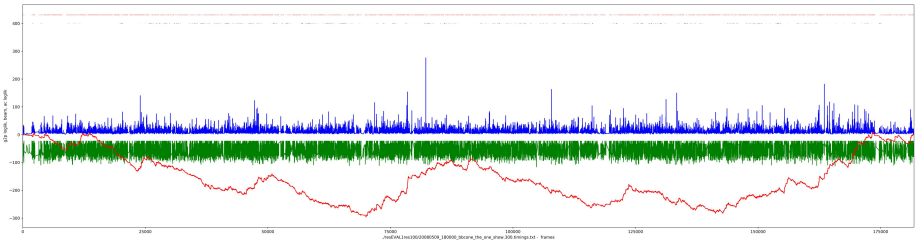
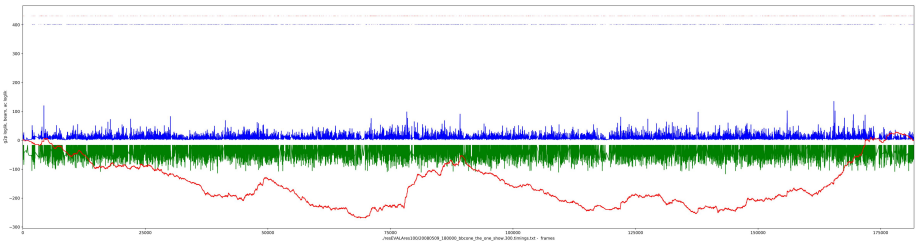


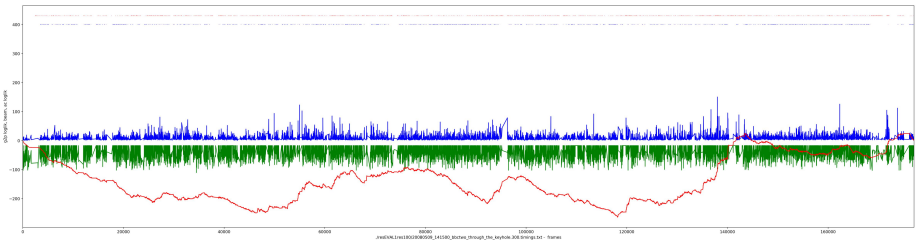
Figure 7.2: Transcript_human 20080509_180000_bbcone_the_one_show



7.3 Observation Two: Beam Close To Zero Is Bad

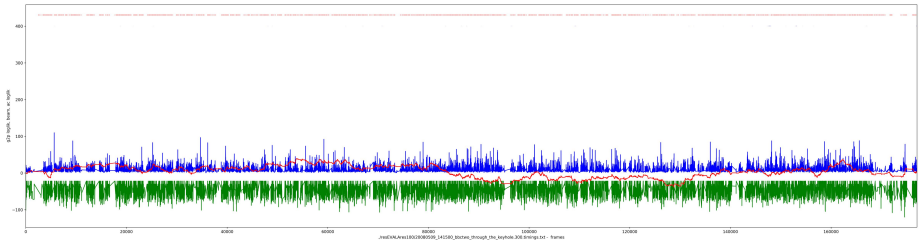
While many tracks performed better when evaluated using transcript_human, the 20080509_141500_bbctwo_through_the_keyhole did not. It behaved as expected with the F-score of 0.6270 for reference and the F-score of 0.0078 for transcript_human. It was therefore the ideal candidate for comparison with 20080509_180000_bbcone_the_one_show (at 100ms boundary). Below you can see what we have plotted out.

Figure 7.3: Reference 20080509_141500_bbctwo_through_the_keyhole



We were trying to find the causes of this anomaly in the plots of all these tracks, yet again, unsuccessfully. What we noticed instead was that, yet again, the worse performing track had the beam value closer to zero. Could this be the theory we were looking for with our Neural Network? Further tests were deemed necessary.

Figure 7.4: Transcript_human 20080509_141500_bbctwo_through_the_keyhole



7.4 Observation Three: Higher The Absolute Value Of Beam The Better

In order to confirm that it is the Beam proximity to zero that causes a low F-score and not simply that higher Beam value means a better score we had to find a track with a high beam value first and then compare it with a worse-performing counterpart. 20080505_180000_bbcfour_the_book_quiz has turned out to be one of the recordings with an overall high beam value. All that was left to do was to plot it out for both reference and transcript_human. Interestingly, 20080505_180000_bbcfour_the_book_quiz was also one of the recordings that have performed better with transcript_human (with the F-score of 0.7302) and opposed to the reference subtitle set (where it gained the F-score of 0.4253 for 100ms boundary). Below are the two plots.

Figure 7.5: Reference 20080505_180000_bbcfour_the_book_quiz

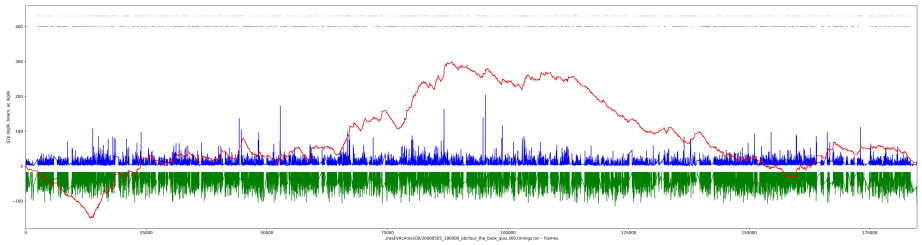
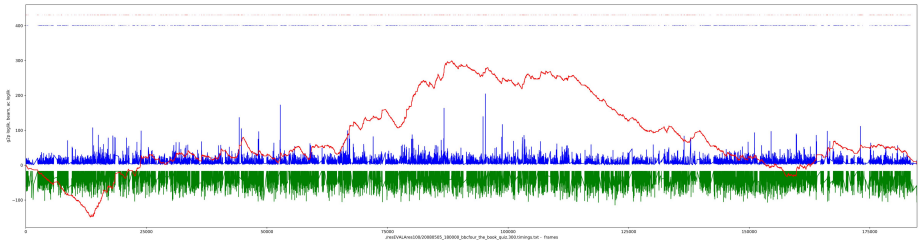


Figure 7.6: Transcript_human 20080505_180000_bbcfour_the_book_quiz



As you can see, our suspicion has turned out to be correct. It is not the higher beam value that worsens the F-score, it is the proximity to the zero value that can signal alignment

failure. Of course, this could all have been a happenstance, a complete coincidence. Our theory needed validation.

7.5 Theory Confirmation

In this section, we try to examine the following theory: "The greater the average absolute value of beam, the greater the F-score". We reuse the data we have gotten from all the previous tests in 5 and use them to confirm or deny our theory.

7.5.1 Scripts Used

In order to test this theory, a special script getbeamvalue.py has been developed. The script takes a folder containing .timings.txt files and subsequently goes through each one of them calculating the average absolute value of beam. The result is then appended to our result file (beamvalue.txt). This evaluation approach goes in keeping with the rest of the work, where we didn't consider the F-score of individual files instead we only focused on the F-score of the data set as a whole.

7.5.2 Results

In the table below results from most of the tests are used to explore the theory that the greater the beam distance the greater the F-score.

Table 7.1: Beam distance to F-score relation

Subtitle set	Phoneme Set	Beam (limitation)	F-Score for 100 ms boundary	Average absolute beam value
reference	New PhnRec EN	10	0.0363	3.5637
reference	New PhnRec EN	30	0.1215	10.6642
reference	New PhnRec EN	50	0.1806	17.5273
reference	Old PhnRec CZ	300	0.0004	34.7537
reference	Old PhnRec EN	300	0.0070	35.7250
reference	New PhnRec EN	100	0.2947	35.7988
transcript_human	New PhnRec CZ	300	0.2765	49.1950
reference	New PhnRec EN	150	0.3651	50.0340
reference	New PhnRec CZ	300	0.2556	53.4472
transcript_align	New PhnRec EN	300	0.1897	57.9574
transcript_orig	New PhnRec EN	300	0.1746	62.0587
reference	New PhnRec EN	200	0.4164	62.8188
reference	New PhnRec EN	250	0.4314	71.4409
reference	New PhnRec EN	300	0.4540	81.0551
transcript_human	New PhnRec EN	300	0.4778	81.6801
reference	New PhnRec EN	350	0.4659	91.1078
reference	New PhnRec EN	400	0.4784	93.2197

7.5.3 Summary: Theory Confirmation

When we look at the 7.1 table we see that with the increase of F-Score, the average absolute beam value also increases. This seems to confirm our theory. When we have tried to solve this problem using neural networks over many repeated attempts, we were mostly unsuccessful (see 6). It was only due to one lucky incident that allowed us this breakthrough in understanding of our text to speech alignment solution. Not only was our solution already capable of training itself on the data that is to be aligned, our alignment solution is capable to successfully self-evaluate without the need for reference data set. This makes our solution truly unique and groundbreaking. For some reason, this self-evaluation does not work in 100% of cases – improper phoneme derivation caused by untrained phoneme recognizer seems to break not only our alignment, but also this evaluation. We can eliminate this problem simply by using a well-trained phoneme recognizer. We plotted our results in the chart below.

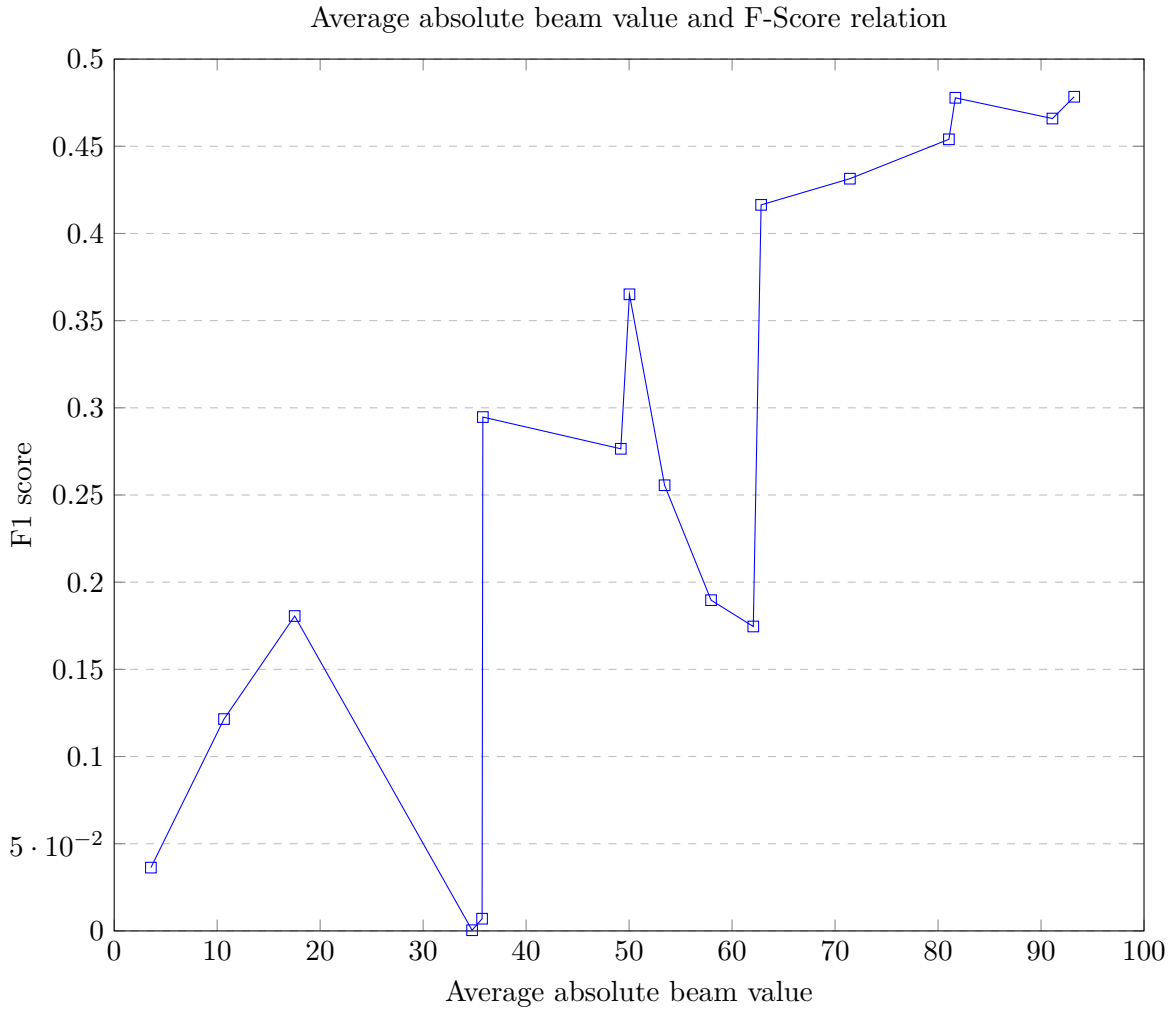


Figure 7.7: Average absolute beam value and F-Score relation

Chapter 8

Conclusion

We have successfully explored the quirks and limitations of a Joint-Sequence Model Viterbi force alignment. We have compared it with other G2P (Grapheme-to-Phoneme) alignment methods. We have focused on pointing out the advantages and disadvantages of each approach. We have tested our JSM alignment solution and we have discovered that the output of our solution relies heavily on phoneme derivation accuracy. To improve the performance of our G2P, new more precise language-independent methods of phoneme derivation should be explored. We have observed that phoneme error influence decreases with the increase of boundary (tolerance) unlike grapheme error which remains proportional to the boundary. We have also discovered that better formatted albeit incorrect grapheme tracks (subtitles) outperform chopped-up reference grapheme track (subtitle set). We have tested our language independent G2P and observed, that more precisely generated phonemes of a different language will outperform language-native imprecise phonemes. We have measured the beam width influence and found that it always follows a logarithmic function. We have attempted to predict final F-score using classification neural network with mixed results. Following that, we have made a breakthrough discovery in observing how the average mean value of beam is directly proportional to the F-score. This allows our solution to assess its own performance without the need for any external data. Our G2P can now use the same data for all three tasks: training, alignment and evaluation. While our G2P doesn't offer the best alignment accuracy there is, its unique qualities and capabilities proudly place it among the titans of text-to-speech alignment.

Bibliography

[1] GitHub - fgnt/nhpylm: Python bindings for a c++ based implementation of the Nested Hierarchical Pitman-Yor Language model.
Retrieved from: https://github.com/fgnt/nhpylm

[2] GitHub - synalp/jtrans: text-to-speech alignment java software.
Retrieved from: https://github.com/synalp/jtrans

[3] Phoneme recognizer based on long temporal context | Skupina zpracování řeči.
Retrieved from: http://speech.fit.vutbr.cz/cs/software/phoneme-recognizer-based-long-temporal-context

[4] Bell, P.: MGB Challenge.
Retrieved from: http://www.mgb-challenge.org/

[5] Bell, P.; Gales, M. J. F.; Hain, T.; et al.: The MGB challenge: Evaluating multi-genre broadcast media recognition. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. Dec 2015. pp. 687–693. doi:10.1109/ASRU.2015.7404863.

[6] Bisani, M.; Ney, H.: Joint-sequence models for grapheme-to-phoneme conversion. *Speech Communication*. vol. 50, no. 5. 2008: pp. 434 – 451. ISSN 0167-6393. doi:https://doi.org/10.1016/j.specom.2008.01.002.
Retrieved from:
http://www.sciencedirect.com/science/article/pii/S0167639308000046

[7] Blair, D. C.: Information Retrieval, 2nd ed. C.J. Van Rijsbergen. London: Butterworths; 1979: 208 pp. *Journal of the American Society for Information Science*. vol. 30, no. 6. 1979: pp. 374–375. ISSN 1097-4571. doi:10.1002/asi.4630300621.
Retrieved from: http://dx.doi.org/10.1002/asi.4630300621

[8] Burget, L.; Schwarz, P.; Agarwal, M.; et al.: Multilingual acoustic modeling for speech recognition based on Subspace Gaussian Mixture Models. In *Proc. International Conference on Acoustictics, Speech, and Signal Processing*, vol. 2010. IEEE Signal Processing Society. 2010. ISBN 978-1-4244-4296-6. ISSN 1520-6149. pp. 4334–4337.
Retrieved from: http://www.fit.vutbr.cz/research/view_pub.php.cs?id=9307

[9] Cerisara, C.; Mella, O.; Fohr, D.: JTrans, an open-source software for semi-automatic text-to-speech alignment. In *Proceedings of the 10th Annual Conference of the International Speech Communication Association - Interspeech 2009*. Brighton,

United Kingdom. September 2009.
Retrieved from: https://hal.inria.fr/inria-00431398

[10] Daumé, H., III: Non-parametric Bayesian Areal Linguistics. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. NAACL '09. Stroudsburg, PA, USA: Association for Computational Linguistics. 2009. ISBN 978-1-932432-41-1. pp. 593–601.
Retrieved from: http://dl.acm.org/citation.cfm?id=1620754.1620841

[11] Fienberg, S. E.: When did Bayesian inference become Bayesian? *Bayesian Anal.*. vol. 1, no. 1. 03 2006: pp. 1–40. doi:10.1214/06-BA101.
Retrieved from: https://doi.org/10.1214/06-BA101

[12] Grézl, F.; Karafiát, M.; Burget, L.: Investigation into bottle-neck features for meeting speech recognition. In *Proc. Interspeech 2009*. 9. International Speech Communication Association. 2009. ISBN 978-1-61567-692-7. ISSN 1990-9772. pp. 2947–2950.
Retrieved from: http://www.fit.vutbr.cz/research/view_pub.php?id=9038

[13] Hannemann, M.; Trmal, J.; Ondel, L.; et al.: Bayesian joint-sequence models for grapheme-to-phoneme conversion. In *Proceedings of ICASSP 2017*. IEEE Signal Processing Society. 2017. ISBN 978-1-5090-4117-6. pp. 2836–2840.
Retrieved from: http://www.fit.vutbr.cz/research/view_pub.php?id=11469

[14] Haubold, A.; Kender, J. R.: Alignment of Speech to Highly Imperfect Text Transcriptions. In *2007 IEEE International Conference on Multimedia and Expo*. July 2007. ISSN 1945-7871. pp. 224–227. doi:10.1109/ICME.2007.4284627.

[15] Hoffmann, S.; Pfister, B.: Text-to-speech alignment of long recordings using universal phone models. 01 2013: pp. 1520–1524.

[16] Hu, J.; Brown, M. K.; Turin, W.: HMM based online handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. vol. 18, no. 10. Oct 1996: pp. 1039–1045. ISSN 0162-8828. doi:10.1109/34.541414.

[17] JOSHI, J.: Prediction of weather states using Hidden Markov model. 12 2012.

[18] Malfrère, F.; Dutoit, T.: High-quality speech synthesis for phonetic speech segmentation. In *EUROSPEECH*. 1997.

[19] Malfrère, F.; Deroo, O.; Dutoit, T.; et al.: Phonetic alignment: speech synthesis-based vs. Viterbi-based. *Speech Communication*. vol. 40, no. 4. 2003: pp. 503 – 515. ISSN 0167-6393. doi:https://doi.org/10.1016/S0167-6393(02)00131-0.
Retrieved from:
http://www.sciencedirect.com/science/article/pii/S0167639302001310

[20] Markov, A. A.; Nagorny, N. M.: *The Theory of Algorithms*. Springer Publishing Company, Incorporated. first edition. 2010. ISBN 9048184533, 9789048184538.

[21] Mehrotra, K.; Mohan, C.; Ranka, S.: *Elements of Artificial Neural Networks*. A Bradford book. MIT Press. 1997. ISBN 9780262133289.
Retrieved from: https://books.google.cz/books?id=6d68Y4Wq_R4C

[22] Novak, J. R.; Minematu, N.; Hirose, K.: Failure transitions for Joint n-gram Models and G2P Conversion.

[23] Perry, J. W.; Kent, A.; Berry, M. M.: Machine literature searching X. Machine language; factors underlying its design and development. *American Documentation*. vol. 6, no. 4. 1955: pp. 242–254. ISSN 1936-6108. doi:10.1002/asi.5090060411. Retrieved from: http://dx.doi.org/10.1002/asi.5090060411

[24] Schwarz, P.: *Phoneme recognition based on long temporal context*. PhD. Thesis. Brno University of Technology, Faculty of Information Technology. 2009. Retrieved from: http://www.fit.vutbr.cz/study/DP/PD.php?id=109

[25] Schwarz, P.; Matějka, P.; Černocký, J.: Towards Lower Error Rates in Phoneme Recognition. In *Proceedings of 7th International Conference Text,Speech and Dialoque 2004*. Springer Verlag. 2004. ISBN 3-540-23049-1. page 8. Retrieved from: http://www.fit.vutbr.cz/research/view_pub.php.cs?id=7646

[26] Schwarz, P.; Matějka, P.; Černocký, J.: Towards Lower Error Rates In Phoneme Recognition. *Lecture Notes in Computer Science*. vol. 2004, no. 3206. 2004: pp. 465–472. ISSN 0302-9743. Retrieved from: http://www.fit.vutbr.cz/research/view_pub.php?id=7483

[27] Serrière, G.; Cerisara, C.; Fohr, D.; et al.: Weakly-supervised text-to-speech alignment confidence measure. In *International Conference on Computational Linguistics (COLING)*. Proceedings of the 26th International Conference on Computational Linguistics (COLING). Osaka, Japan. December 2016. Retrieved from: https://hal.archives-ouvertes.fr/hal-01378355

[28] Stevenson, A.: *Oxford Dictionary of English*. ISBN 9780199571123. Retrieved from: http://www.oxfordreference.com/view/10.1093/acref/9780199571123.001.0001/acref-9780199571123

[29] Teh, Y. W.: A Hierarchical Bayesian Language Model Based on Pitman-Yor Processes. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*. ACL-44. Stroudsburg, PA, USA: Association for Computational Linguistics. 2006. pp. 985–992. doi:10.3115/1220175.1220299. Retrieved from: https://doi.org/10.3115/1220175.1220299

[30] Viterbi, A.: Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*. vol. 13, no. 2. April 1967: pp. 260–269. ISSN 0018-9448. doi:10.1109/TIT.1967.1054010.